Incremental Transitivity Applied to Cluster Retrieval

Yaser Hasan[1], Muhammad Hassan[2], and Mick Ridley[1]
[1]Computing Department, University of Bradford, UK
[2]Computer Science Department, Zarqa Private University, Jordan

Abstract: *Many problems have emerged while building accurate and efficient clusters of documents; such as the inherent problems of the similarity measure, and document logical view modeling. This research is an attempt to minimize the effect of these problems by using a new definition of transitive relevance between documents; i.e., adding more conditions on transitive relevance judgment through incrementing the relevance threshold by a constant value at each level of transitivity. Proving the relevance relation to be transitive, will make it an equivalence relation that can be used to build equivalence classes of relevant documents. The main contribution of this paper is to use this definition to partition a set of documents into disjoint subsets as equivalence classes (clusters). Another contribution is by using the incremental transitive relevance relation; the traditional vector space model can be made incrementally transitive.*

Keywords: *Clustering, equivalence class, information retrieval, incremental transitivity.*

Received May 10, 2007 ; accepted December 26, 2007

1. Introduction

Cluster retrieval was proposed as a solution for the problem of high scale diversity among large collections of topics [6], which compromise a large number of documents. It is possible that documents which belong to different topics are constructed from the same set of words (vocabulary), as a consequence, they can share some words in common that have different meanings, or conversely share the same semantics but are represented by different terms. This kind of representation leads to some ambiguity when documents are being modeled, since most models use index terms as bases for document representation [15]. The choice of a similarity function will also affect the properness of relevance judgment, and so affecting the *theoretical soundness*[1] condition of the clustering method. Consequently, this overlapping among documents representations will lead to wider overlaps among topics. So for better topic determination and so better structuring of the collection into topics and sub-topics, the ambiguity among documents must be resolved. Better retrieval efficiency is reached when user needs are matched with documents grouped in the same context; i.e., within the same cluster [18], rather than matching with documents which may be classified under more than one topic. Some studies rely on overlapping between topics and return relevant clusters (topics but not documents) in response to user requests, as in [20] for example.

This paper presents a new approach to solving this problem of ambiguity using the transitive relevance interrelationships between documents. A new definition of transitivity is proposed to make similarity, computed by the traditional vector space model, *incrementally* transitive. Then sub collections are more likely to have the same semantics as clusters. That is likely to minimize the effects of ambiguity resulting from the drawbacks of the traditional model, and helps to strengthen the relevance judgment among groups of documents.

Our aim is to provide more accurate aggregation of documents into clusters that fulfill the requirements of retrieval efficiency, and the requirements of other types of applications where accuracy is very important, for example when a peer node in a peer-to-peer IR system needs to inform other nodes about what type of information, or what topics, it contains, as in [2].

The proposed method applies the following scenario of transitivity:

A. Use the cosine similarity to judge direct relevance between an arbitrary chosen documents according to some threshold.
B. Add relevant documents to the same cluster.
C. Use these relevant documents as parents to sub-trees to find transitive relevant documents after incrementing the threshold by a constant fraction.
D. Use the newly added documents, again, as roots and increment the threshold.

[1] It was reported by Rijsbergen in [13]:

- The method produces a clustering which is unlikely to be altered drastically when further objects are incorporated, i.e. it is stable under growth.
- The method is stable in the sense that small errors in the description of the objects lead to small changes in the clustering.
- The method is independent of the initial ordering of the objects.

E. Repeat to the next level of transitivity until the threshold reaches one, or no more relevant documents are found.
F. Repeat the same process until all possible clusters are found.

Once the relevance relation is proved to be *incrementally* transitive, it is easy to show that relevance is an equivalence relation; equivalence relation can partition the set of objects into *disjoint* subsets or *equivalence classes*. As known from Algebra, see [17] for example, each subset can be represented by a member of the equivalence class. The elements of the subset are tightly related to this representative and far from other representatives of other equivalence classes. Creating clusters that meet the conditions of equivalence classes will lead to clusters better meeting rijsbergen cluster hypothesis: "closely associated documents tend to be relevant to the same requests," and the theoretical soundness of the clustering method [13].

Many methods have been proposed to solve the problem of representation ambiguity; Latent Semantic Indexing (LSI) [5] used space reduction to eliminate *noise* from documents logical views. Cluster retrieval used interrelations between documents to build semantic blocks of documents [6, 12, 14, 13, 20]. Query expansion is being used to enhance the users' needs and to define the actual context of terms entered by users found in [6, 12]. Term co-occurrence is also used for more context widening while matching user requests [10]. Throughout these models, similarity between documents, or the context of terms is used to judge relevance between documents and users' requests, and build semantically related groups.

The rest of this paper is organized as follows. Section 2 presents the theoretical background of incremental transitive relevance. Section 3 introduces the proposed algorithm that relays on the definition of transitivity discussed in section 2, including complexity analysis. Section 4 contains the experimental test of the algorithm and results. Finally, section 5 concludes our work.

Throughout this paper, *incremental* transitivity and transitivity are used as synonyms. The term *direct transitivity* is explicitly used to indicate the usual meaning of transitivity.

2. Incremental Transitivity

Definition: Incremental transitive relevance.
Given a collection of documents *D, and* the relevance relation $\Re$ defined on *D,* such that

$$\Re=\{(x,y): sim(x, y) \geq \delta, \forall documents\ x, y \in D\} \quad (1)$$

where δ is the relevance threshold, and $sim(x,y)$ is the similarity measure between x and y. Given d_1, and $d_3 \in D$ are two normalized vectors, each represents a document in the collection D, and $sim(d_1,d_3) < \delta$, d_1 and d_3 can have *incremental transitive relevance* if there exists a document $d_2 \in D$ such that $sim(d_1, d_2) \geq \delta$, and $sim(d_2, d_3) \geq \delta+\varepsilon$, where ε is a constant real number; i.e., if *we have* $(d_1, d_2) \in \Re$, *and* $(d_2, d_3) \in \Re$ implies that $(d_1, d_3) \in \Re$ with threshold $\delta+\varepsilon$ then $\Re$ is the incremental transitive relevance relation.

The relation $\Re$ could be then defined as a virtual (not mathematically proved) equivalence relation to form *virtual equivalence classes* of documents as clusters. It is easy to show that $\Re$ is reflexive (each document is relevant to itself), symmetric (relevance is commutative), and if it is incrementally transitive then it is equivalence relation.

Since "Clustering" of documents is the grouping of documents into distinct classes according to their intrinsic (usually statistical) properties,[6] and it is known from algebra that the set of equivalence classes defined by an equivalence relation is a partition of the set of objects, [17] then $\Re$ could be used to build clusters most likely to represent disjoint topics.

Incremental transitivity is proposed as incrementing the threshold value to get a sequence of incremental threshold:

$$\delta_i = \delta^\circ + i \cdot \varepsilon \quad (2)$$

where δ° is the initial relevance threshold, i is the transitivity level, and $\varepsilon \in [0, 1)$.

Expanding this definition to $d_1, d_2, \ldots, d_n$, of $n-1$ incremental transitive relevance levels, then any document vector $d_i \in D$, $i = 1, \ldots, n$, that has direct relevance with d_{i-1} can have transitive relevance with d_1 if $sim(d_1, d_{i-1}) \geq \delta^\circ + (i-1) \times \varepsilon$. practically, when using index terms to model documents in Vector Space Model (VSM); for example, we can find (by a human judge) two relevant documents without having enough common index terms between them [3], and so we cannot (automatically) get the similarity value that makes d_1 and d_3 relevant, and so we couldn't declare relevance through direct transitivity; i.e., VSM is not transitive.

The increment ε is applied to the threshold value to eliminate the *error* in relevance judgment, since it is known in the literature of information retrieval that index terms are not completely disjointed or independent [15]. The traditional vector space model assumes the independence among index terms. For it to be transitive (we believe) it is necessary to increase the threshold by a value which is equivalent to the error of the similarity measure, in this paper the selected value of the ε is the Standard Deviation (SD) among similarity values computed by the cosine test between documents, which is explained in the next section.

3. Clustering Algorithm

The following algorithm uses incremental transitive relation to build clusters of documents. This algorithm assumes the existence of a document-to-document

similarity matrix, and a predefined initial threshold value (δ°) for relevance judgement.

3.1. Choosing Initial Relevance Threshold (δ°)

The threshold choice will affect the initial relevance judgment accuracy, and so for it to be more accurate the initial threshold should be selected taking into consideration the statistical characteristics of the collection of documents, and the degree of accuracy needed; i.e., the number of clusters, and the distribution of documents onto clusters. A variant of the dynamic threshold [6] value will be used to determine relevance. Statistical properties that represent interrelations between documents are: the AVeraGe (AVG) of similarity values, and the SD among them, where SD value is selected to be the threshold increment (ε). Where the AVG similarity is computed by the equation:

$$AVG = \frac{\sum_{i=1}^{n-1}\sum_{j=1}^{n} sim(d_i, d_j)}{n(n-1)}, \; sim(d_i, d_j) \geq \delta^\circ \qquad (3)$$

the considered similarity values are those having $sim(d_i, d_j) \geq \delta^\circ$, since the only loaded similarity values that are greater than or equal to the initial threshold. The SD is given by equation (4) and calculated for the similarity values as in equation (3).

$$SD = \sqrt{\frac{\sum_{i=1}^{n-1}\sum_{j=1}^{n} (sim(d_i, d_j) - AVG)^2}{n(n-1)}} \qquad (4)$$

```
procedure CreateClusters;
determine α value;
determine ε value;
let δ°=AVG_similarity + α × ε;
for each document dj, j=1,.. N-1 do
    consider dj as a root node of a sub-tree;
    for each document di, i=j+1, .., N do
        read from DocToDocMatrix entries that
        have sim(dj, di) ≥ δ°;
        add di as a child to dj;
    next i
    if i=j+1 then    // no relevant documents found
        delete dj sub-tree;
    end if
next j
let current sub-tree = the first sub-tree (d1);
while not reaching the last sub-tree node do
Threshold=δ°; // initialize the threshold
add children of current sub-tree to the queue;
pull a node from queue, call it Ptr;
while (queue is not empty) and (Threshold<1) do
    let Threshold=δ+level(Ptr)*ε;
    if (sim(Ptr, dj) ≥ Threshold) and
    (Ptr is a root of a sub-tree) then
        attach Ptr sub-tree as a child of dj;
        add Ptr children to the queue;
        delete Ptr sub-tree; // from the zero level
    end if
    pull a node from the queue, assign it to Ptr;
end while
clear the queue;
get next sub-tree; // j is the id of the document that
end while              // represent the next remaining
end procedure.     // sub-tree
```

Figure 1. Clustering algorithm using incremental transitivity.

The following equation will be used for the initial threshold determination:

$$\delta^\circ = AVG + \alpha \times \varepsilon \text{ , } \alpha = 1,2, .. ,r \qquad (5)$$

where ε= SD of similarity values calculated by equation 3, and α is a constant integer selected such that $\delta^\circ \leq 1-\varepsilon$. The α value determines the initial relevance threshold (δ°), so higher values of α leads to selection of only documents that are strongly relevant to the root document (each cluster is a sub-tree in a hierarchical structure), and these documents could be parent nodes of incremental transitive documents at higher levels. Each sub-tree is considered as a candidate cluster, because they may attach to other cluster as we will see next in example 1.

3.2. The Algorithm

The clustering algorithm, Figure 1, uses the idea of previous sections to cluster a set of documents, where the doc-to-doc matrix is a prerequisite, and the initial threshold is calculated by using equation 4, lines (2-4). The second part of the algorithm (lines 5-15) loads only those documents that have similarity values greater than the initial threshold (δ°), adding them as children to the pre-read document (d_j). If d_j has no similar documents (according to the selected δ°) then remove its sub-tree from the zero-level; i.e., don't consider it as a root to any candidate cluster, (lines 12-14). For example, in the Reuters21578 test collection used in this study for evaluation, the ratio of similarity values that were greater than (δ°) was about 2.2% of the overall matrix entries, for α=1.

The second step of the algorithm (lines 16-33) starts at the first sub-tree, as the current sub-tree, putting all of its children in a queue, this queue is used to test all siblings in the same level before moving to any of their children in the next level, and so giving counterpart documents in the same level the priority for transitive relevance test before documents in higher levels, just like the breadth-first tree search algorithm.

For every new cluster (represented by a sub-tree); initialize the threshold, pull an item from the queue (assign it to *Ptr*, as in line 20), calculate the threshold value by adding the product of epsilon and the level of the pulled item; i.e. to increment the threshold value by a multiple of epsilon dedicated for this level, according to the threshold equation (1). If the similarity between *Ptr* and its root of the sub-tree (d_j) is greater than the new calculated threshold, and the document represented by *Ptr* is a root of another sub-tree, then remove its sub-tree, and attach its children to *Ptr*, those children are new discovered relevant documents through incremental transitivity, since they don't have direct relevance to d_j. The attached children are added to the queue for future testing, since they may represent documents that are roots of other sub-trees. Keep pulling the queue, each time assign the pulled node to *Ptr*, and repeat as above, until the queue is

empty; i.e., no more documents are found with incremental transitive relevance to d_j.

Switch to the next remaining sub-tree, and repeat the above scenario until no more sub-trees remain. Since document existence is checked only within the same cluster (which is represented as a sub-tree), then a document could be attached to more than one cluster; i.e., a document could be shared by more than one topic, as in [3]. Each time the algorithm attaches a sub-tree it means that it decrements the number of sub-trees (candidate clusters), from the zero-level, at the end of the process the remaining zero level sub-trees are the final clusters. The next example gives more details about the process.

Example 1: In this example, Figure 2 represents a part of an initial structure of cluster hierarchy consisting of the zero and 1-level similarity tree. Let the initial threshold $\delta°=0.15$, and $\varepsilon=0.125$. It can be seen that all the nodes in the 1-level have similarity$\geq \delta°$, since the only loaded similarity values that are above the initial threshold. Documents: d_5 and d_6 are not represented as nodes in the 0-level because none of the remaining documents have similarity $\geq \delta°$ with them. When applying the algorithm, starting by d_1 sub-tree, the queue content becomes: (d_4, d_5, d_6), $\delta_1=0.275$, pull d_4 (since it has $sim \geq \delta_1$), remove its node in the level-0 (d_4, L=0, S=1), add its child, d_7, to the queue (queue contents are then: d_5, d_6, d_7) after attaching it to the node (d_4, L=1, S=0.29) in level-1 which belongs to d_1 sub-tree. Nodes *d5* and *d6* have no children, the queue content is (d_7), pull d_7, it has a child d_8, attach it as a child in level-3 since $sim(d_7, d_8) \geq \delta_2=0.4$.

Restore the initial value of threshold, and the same operations are applied to the sub-tree (d_2), it has d_3 child of similarity greater than δ_1, so attached to d_2 sub-tree using the same method. Finally the hierarchy will be as in Figure 3. The resulted clusters are: $C_1=\{d_1, d_4, d_5, d_6, d_7, d_8\}$, and $C_2=\{d_2, d_{17}, d_3, d_{10}, d_{14}\}$.

3.3. Cluster Refinement

After generating clusters, it can be seen that many clusters will have a smaller sizes; i.e., having sizes less than the average cluster size (average number of documents in a cluster), resulted from the first pass (CreateClusters). Ignoring these clusters will harm the overall recall ratio; i.e., some documents may not included in any cluster, and so when responding to the users requests, using cluster-retrieval, none of these documents will be retrieved as a result of any user query, so decreasing the ratio of retrieved relevant documents, this will result in less accurate clusters. To avoid the effect of having lower recall ratio, a *refinement* algorithm (as a second pass) is being used, steps of this process are:

A. Determine number of refinement cycles; N.
B. Do CreateClusters();
C. For i = 1 to N do
C.1. *Divide Epsilon by i;*
C.2. Recalculate Threshold;
C.3. Do CreateClusters();
Next i;
D. SaveRefinedClusters()

Refinement objectives are achieved by depressing the conditions imposed on relevance judgement while creating clusters and so more documents will be attached to a closer relevant sub-tree. The refinement is a second pass of the algorithm following the first pass; i.e., create-clusters. Empirical study shows that refined incremental transitive clusters have intermediate quality between not refined incremented transitive clusters, and not refined transitive clusters created without increments on threshold.

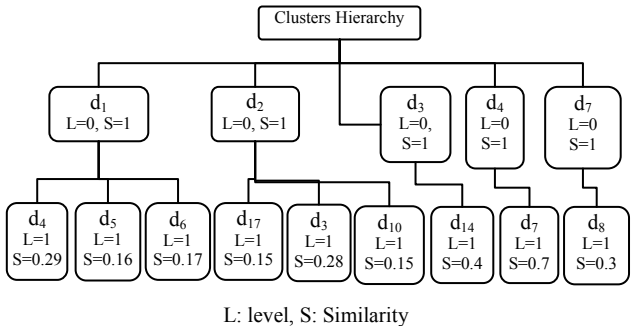

Figure 2. The initial structure of cluster hierarchy of example 1.

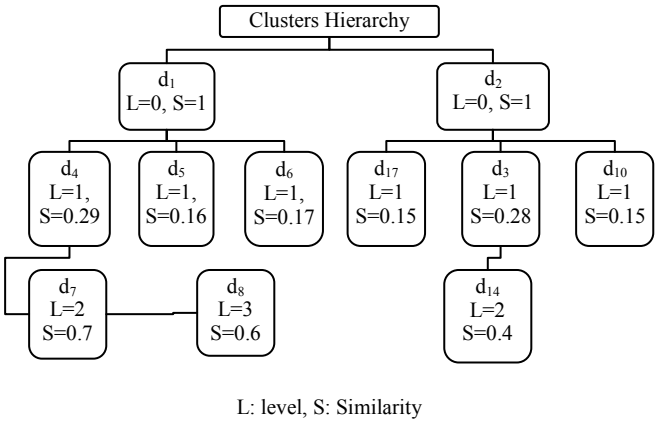

Figure 3. The final structure of cluster hierarchy in example 1.

3.4. Algorithm Complexity

The only entries who involved in the clustering process are those having similarity greater than or equal the initial threshold.

Let N be the number of documents, and U_s is the maximum number of similarity values that are above the initial threshold in a column of doc-to-doc matrix, then we can have the following as given:

- Number of entries involved are bounded above by: $(N-1).U_s$.
- The probability of having similarity $\geq \delta_\circ$ is:
  - $P(Sim \geq \delta_\circ) \leq ((N-1).U_s)/N$, done while reading the matrix.
- As the threshold is incremented, fewer entries will be available to the next step.

- As we have $\delta_\circ = AVG + \alpha \cdot \varepsilon$, and most similarity distributions are skewed to the left; see Figure 2 for the distribution of Reuters21578 test collection.
- These distributions will be considered bounded above by the function $1/x$.

So the summation:

$$N \sum_{i=1}^{N-1} P_i \le N. \int_{1}^{N-1} \frac{1}{x} dx \qquad (6)$$

and so the complexity is bounded above by: $N \cdot log(N-1)$, in fact it is much smaller since the number of tested sub-trees is becoming smaller as attachments are taking place.

4. Empirical Study

The proposed clustering algorithm has been applied to selected set of documents derived from the Reuters21578 test collection; it is implemented as a complete hierarchical clustering algorithm with some implications, in which the document-to-document matrix is not completely loaded into memory; as explained in section 3. A tree structure is used to represent the matrix, where the actual transitivity levels are implemented as tree levels, the direct relevant documents to each sub-tree root has a level=1, and all the higher transitive levels have incremental transitive relevance to their parents.

4.1. Indexing

A document of the Reuters21578 set is selected if it has a title and a body text; Table 1 summarizes the statistics of the selected documents used in this study.

The VSM is used to build the logical view of the selected documents, porters' algorithm is applied to index term stemming, and weights were calculated according to the equation: [1]

$$w_{i,j} = tf_{i,j} \cdot idf_i, \quad where tf_{i,j} = \frac{f_{i,j}}{maxf_j}, and idf_i = log\frac{N}{n_i} \qquad (7)$$

where $f_{i,j}$ is the physical occurrence of term i in document j, and max frequency is the maximum frequent term in document j. N is the total number of documents and n_i is the number of documents contain this term.

Table1. Reuters21578 statistics.

Number of documents	18650
Average similarity	0.009
Standard deviation (ε)	0.0313
Initial threshold	0.0403
Number of index terms	30061
Number of Similarity values >= average, α=3.	3811191 ≈ 2.2% of total values
Total Number of values	173901925

Terms occurring in the title of each document are given twice the weight of other terms, since it has been shown to be more efficient to give heading terms more rank while indexing. [3, 9, 16] Each document is represented by a normalized vector of weights previously calculated by equation 7, and the cosine Similarity function, as shown in equation 8 is used to determine relevance between documents.

$$Sim(\vec{d}_1, \vec{d}_2) = \sum_{i=1}^{n} d_{1i} \cdot d_{2i} \qquad (8)$$

where $\vec{d}_1$ and $\vec{d}_2$ are normalized document vectors.

4.2. Evaluation

In order to evaluate incremental transitive clustering, the method is compared to a transitive clustering with constant threshold, without increments; i.e., ε=0, and a third method with refinement applied to the incremental transitive clustering.

The three methods are implemented several times using different values of the initial threshold (δ°), gained by varying the value of α (from α=3 to α=10), clusters are considered better if they adhere to the following criteria:

- Gathering higher number of documents that share the same topic, as previously determined by human experts.
- If the percentage of larger clusters; is higher, with smaller standard deviation between clusters sizes.

This is important because it allows the algorithm to assign documents to the right topic(s), and discover more transitive relevant documents sharing the same topic without having enough common terms. More formally, if ($d_1 \Re d_2$), and ($d_2 \Re d_3$) but not ($d_1 \Re d_3$), where ($d_1 \Re d_2$) means that d_1 has direct relevance with d_2, and the algorithm decides ($d_1 \Re d_3$) through transitivity, then there are two possibilities, either:

- d_1 and d_2 belong to a topic T_1, but d_2 and d_3 have $Sim(d2, d3) \ge \delta° + i \times \varepsilon$, because they share another different topic T_2, then the decision making d_1, d_2, and $d3$ belong to T_1 may not be accurate, resulting in faulty clusters, or:
- It is successfully in assigning d_1 and d_3 to the same cluster because they share the same topic, but this was not detected previously because of indexing faults (mainly caused by the ambiguity of term independence assumed by VSM) or inaccurate similarity measurement.

Cluster size has been determined by different researchers to affect clustering performance; [7] considered that cluster performance is strongly related to cluster size. [4] considered cluster size when determining the purity of a class (topic) within a cluster. And [8] concentrate on cluster size determination to affect cluster performance.

Evaluation procedure: Two variables; cluster size, and the ratio of documents in a cluster that share the same topic(s), were tested for the three methods: constant transitivity (without threshold increments i.e., ε=0), a second time with increments applied to the

relevance threshold; i.e., ε=SD of similarity values, and the third with refinement applied to clusters created using incremental transitivity.

Reuters21578 documents were assigned topics by David D. Lewis from AT&T Labs-Research. The term "topics" is one of the following:

- <TOPICS> There are 135 different topics.
- <PLACES> Total of 175 different possible places were assigned.
- <PEOPLE> 267 possible values.
- <ORGS> 56 values.
- And <EXCHANGES> 39 values. Each of these options may have more than one value; each value is added as a <D> tag.

A Java program is built for the purpose of this research, where all the steps of indexing, clustering, and evaluation were implemented.

Evaluating topic occurrence is done by constructing a topic matrix for each cluster; see Figure 4, where each document has a row vector of binary values; one if the document was assigned the topic which heads the desired column, otherwise zero. Topic sharing ratio is calculated by using equation 9:

$$Tr_{ij} = \frac{\sum_{i=1}^{m} T_{i,j}}{Cs_j} \quad (9)$$

where Tr_{ij} is the ratio of topic i in cluster j, Cs_j is size of cluster j, $T_{ij} \in \{0, 1\}$, and m is the total number of topics, for Reuters21578 m=681. Better clusters are considered to have larger values of Tr_{ij}.

4.3. Results and Discussion

Cluster size: For the two methods where incremental transitivity is used (with, and without refinement), as α increases the number of clusters having size above the average clusters increases; i.e., larger values of α result in more cohesive clusters, since relevance judgement depends on larger initial threshold, but this is not the situation when no increments are applied, see Figure 5. From this Figure it is clear that incremental transitivity best improves the number of above-average sized clusters, and at the same time decreases the average cluster size; as shown by Figure 6. Refined clusters have intermediate quality between incremented (ε=SD) and non-incremented (ε=0) transitive clusters.

Topics / Documents	T_1	T_2	…..	T_m
d_1	1	0		0
d_2	1	1		0
d3	1	0		1
Tr_{ij}	100%	33%		33%

Figure. 4. Example of cluster-topic evaluation matrix.

Differences among clusters sizes (detected by the standard deviation of cluster size) decreased, which give more appropriateness to resulted clusters, see Table 2. Better cluster sizes are gained when using incremental transitivity, even for smaller values of the initial threshold (smaller α), which proves the importance of using incremental transitivity to discover indirect relevance among documents without having to increase the initial threshold. Higher initial threshold values impose more conditions on similarity values, which could be loaded, and so the number of documents involved in the clustering process could be smaller, see Table 4, for the relation between the initial threshold and the similarity values loaded.

Topic occurrence Ratio (Tr_{ij})*:* Equation 8 is used to calculate the ratio of documents having a certain topic shared among them in a cluster. It can be noticed that the number of topics shared by at least 97% of documents in each cluster when using incremental transitivity is better than in the other two methods; it reaches, for example, 29 topics shares at α=6, see Table 3, while in the case of no increments; i.e., ε=0, a similar value could be reached at α=9, Figure 8 presents a comparison between clusters built by using the three methods.

The same result could be found when comparing the number of shared topics between 50% of documents in clusters; the incremental transitive method reaches 289 shared topics at ∝=4, see Table 5, while in case of constant threshold transitive method; i.e. when ε=0, a similar number could be reached at ∝=9.

Table 2. Cluster size statistics for two values of α.

	α=3			α=7		
	AVG[1]	SD[2]	>=AVG[3]	AVG[3]	SD[2]	>=AVG[3]
ε=SD	174	766	164	31	165	485
ε=0	292	1454	25	40	447	133

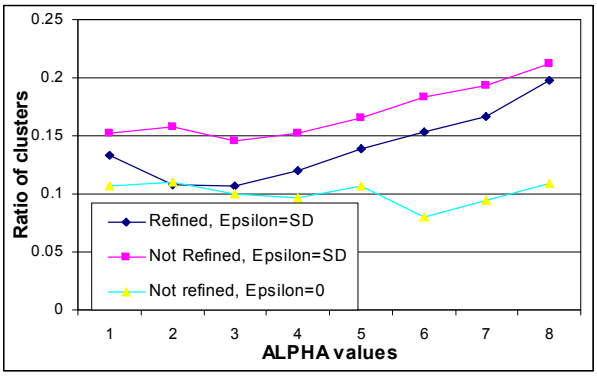

Figure 5. Ratio of clusters having sizes above the average cluster size for different values of α.

There is considerable evidence that better clusters are produced at lower levels of initial threshold when using incremental transitivity; i.e., more relevant documents are discovered. Creating better clusters at lower levels of initial threshold strengthen the correctness of the assumption, which is: the cosine similarity measure defined of the traditional vector space model gains transitive property when increments

[1] Average cluster size
[2] Standard deviation of cluster size values
[3] Size ≥ average cluster size.

are applied to the threshold, and gives more evidence to the relevance relation $\mathcal{R}$ being the equivalence relation. The other benefit of having better clusters at lower initial threshold is the lower probability of losing documents; i.e., a document is lost when it is not assigned to any cluster. Some of doc-to-doc matrix columns may not be read, because none of the documents that follow a given document, has similarity above the initial threshold with that document, and so if it does not have a similarity with any document, then it will not appear to exist in the cluster hierarchy, so it will not be assigned to any cluster, and so it is lost.

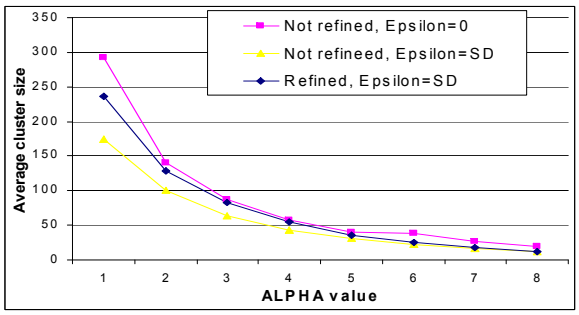

Figure 6. Average cluster size for different values of α.

Table 3. Topics shared by at least 97% of documents in a cluster.

Alpha Method	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
NR[2],ε=SD	20	34	12	29	134	221	304	378
R, ε=SD	15	15	16	29	53	85	143	250
NR,ε=0	5	8	11	1	16	16	29	53

As the ratio of lost documents decreased, the chance of more relevant documents sharing a cluster increased, and so the ratio of retrieved relevant documents also increased; gaining better recall ratio. The distribution of clusters that have at least 50% of their documents sharing at least one topic is better when using incremental transitive relevance than in the two other methods for all possible ratios. This means that the probability of a cluster representing a topic is better when using incremental transitivity regardless of the degree of accuracy needed; Figure 7 gives a comparison of distributions of the three methods.

The effect of increment value ε: Following in this section is a repetition of the previous experiments applied using a fixed value of α; i.e., fixed initial threshold, and a variable ε, each time is applied with smaller value of the increment ε.

Table 4. Possible losses as α increased.

α	**δ**	**Unread columns[1]**	**Total number of similarity values read from doc-to-doc matrix**
3	0.103	252	3811191
7	0.228	1677	808281
10	0.322	3796	307819

The two variables: cluster size, and topic sharing are used for testing the effect of the increment value on clusters performance. The increment value will be replaced by a fraction of SD value. The experiment is repeated for 10 values of ε, each time a tenth of the original value (which was the SD) is subtracted, the first experiment uses a value of ε=SD, the second ε=0.9×SD, …, 0.1×SD, all were done using incremental transitive threshold without refinement, and the selected value of alpha is (α=10). Figure 9 shows the effect of increment value used on clusters sizes, clusters became larger when using smaller values of epsilon, since these smaller values decrease the conditions on the incremental transitive relevance and so more documents will share the same cluster, but the chance of having the wrong relevance judgement became larger. And so the ratio of documents that share a topic in a cluster is getting smaller when ε value decreases, see Figure 10.

As a consequence, smaller increments lead to creation of clusters with less accuracy. As ε approaches zero, the similarity defined on traditional vector space model loses its transitive property. And so the relevance relation ($\mathcal{R}$), defined on documents modelled using VSM, is no longer transitive; i.e., not considered as a good partitioning criteria; not equivalence relation, and clusters built using this relation are not the right part ions of the collection; i.e., not equivalence classes.

Table 5. Topics that are shared among 50% of documents for all clusters (two documents may share more than one topic).

	Alpha Value (α)							
METHOD	3	4	5	6	7	8	9	10
NR,ε=SD	197	289	314	416	660	904	1171	1372
R, ε=SD	110	128	165	243	356	501	686	1019
NR, ε=0	35	61	82	90	165	164	243	364

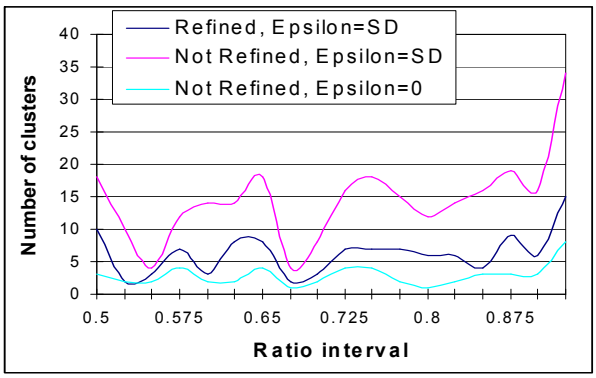

Figure 7. Topic sharing ratio distribution for alpha=4.

5. Conclusion

The main conclusion of this study was: clusters made by incremental transitivity were better formed than those without applying these increments, they are better in two ways:

[1]Documents don't have similarity greater than δ with any document next to its column in the doc-to-doc matrix.
[2] NR: No Refinement, R: With Refinement.

- Getting clusters having sizes greater than the average with less diversity; i.e., more relevant documents were discovered without having enough common terms.
- The ratio of topics shared by most of the documents belonging to a cluster (tested for 97%, and 50% of documents in a cluster) was greater.

The use of increments of threshold with transitive relevance could be helpful in finding hidden relevance, or it could be useful in overcoming problems of ambiguity caused by indexing or by similarity measuring techniques when the traditional VSM is used, this make it more transitive; and so the relevance relation ($\Re$) is closer to being an equivalence relation, and clusters built using this relation are closer to equivalence classes.

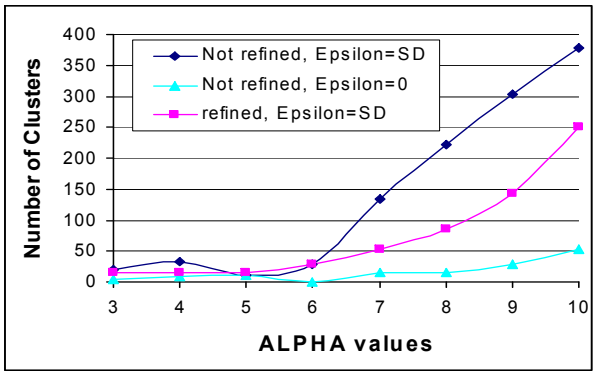

Figure 8. Clusters having 97% of their documents sharing at least one topic for different values of initial threshold determined by α.

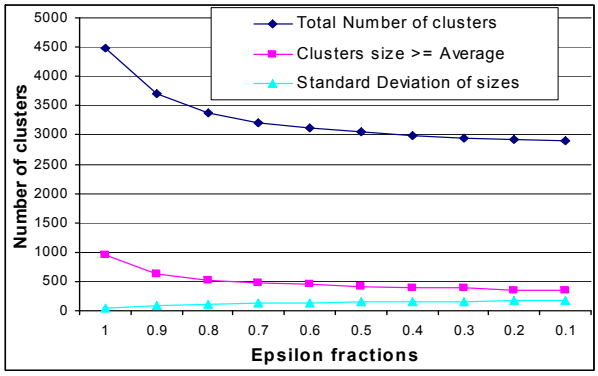

Figure 9. Clusters properties calculated for different values of the increment (ε).

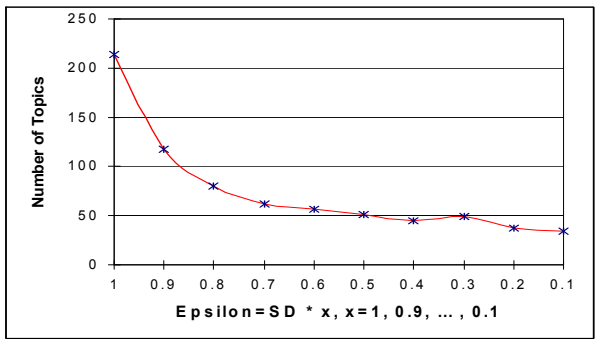

Figure 10. Number of topics shared by ≥97% of documents in a cluster for different values of the increment (ε).

References

[1] Baeza-Yates R. and Ricardo-Neto B., *Modern Information Retrieval*, Addison Wesley, 1st edition, 1999.

[2] Chen Y., Xu Z., and Zhai C., "A Scalable Semantic Indexing Framework for Peer-to-Peer Information Retrieval," *SIGIR 2005 Workshop: Heterogeneous and Distributed Information Retrieval*, pp. 33-40, 2005.

[3] Chunqiang T., Zhichen X., and Sandhya D., "Peer-to-peer Information Retrieval Using Self-Organizing Semantic Overlay Networks," *in Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 175-186, 2003.

[4] Crestani F. and Wu S., "Testing the Cluster Hypothesis in Distributed Information Retrieval," *Information Processing and Management*, vol. 42, pp. 1137-1150, 2006.

[5] Deerwester S., Dumais S., Furnas G., Landauer T., and Harshman R., "Indexing by Latent Semantic Analysis," *American Society for Information Science*, vol. 41, 1990.

[6] Greengrass E., "Information Retrieval: A Survey," 30 November, 2000**,** available on-line www.csee.umbc.edu/cadip/readings/IR.report.120600.book.pdf, 2006.

[7] Hearst M. and Predersen J., "Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results," *in Proceedings of the ACM/SIGIR*, pp. 76-84, 1996.

[8] Huijsmans D. and Sebe N., "Extended Performance Graphs for Cluster Retrieval," in *Proceedings of the IEEE International Conference of Computer Vision and Pattern Recognition (CVPR '01)*, pp. 26-31, 2001.

[9] Kolcz A., Prabakarmurthi V., and Kalita J., "Summarization as Feature Selection for Text Categorization," *in Proceedings of the 10th International Conference on Information and Knowledge Management*, Atlanta, pp. 365-370, 2001.

[10] Kontostathis A. and Pottenger W., "A Framework for Understanding LSI Performance," *Information Processing & Management*, vol. 42, no. 1, pp. 56-73, 2006.

[11] Kuo J. and Chen H., "Cross-Document Event Clustering Using Knowledge Mining from Co-Reference Chains," *Information Processing & Management*, vol. 43, no. 2, pp. 327-343, 2007.

[12] Na S., Kang I., Roh J., and Lee J., "An Empirical Study of Query Expansion and Cluster-Based Retrieval in Language Modelling Approach," *Information Processing and management*, vol. 43, no. 2, pp. 302-314, 2007.

[13] Rejsbergen C., *Information Retrieval*, Butterworth, London, 2nd edition 1979.

[14] Roony N., Patterson D., Galushka M., and Dobrynin V., "A Scaleable Document Clustering

Approach for Large Document Corpora," *Information Processing and management*, vol. 42, pp. 1163-1175, 2006.

[15] Sabharwal C., Hacke K., and Clair D., "Formation of Clusters and Resolution of Ordinal Attributes in ID3 Classification Trees," *in Proceedings of the 1992 ACM/SIGAPP Symposium on Applied Computing: Technological Challenges of the 1990's*, pp. 590-597, 1992.

[16] Shen D., Chen Z., Yang Q., Zeng H., Zhang B., and Lu Y., "Webpage Classification through Summarization," *in Proceedings of the 27th ACM SIGIR Conference*, pp. 242-249, 2004.

[17] Walker E., *Introduction to Abstract Algebra*, New Mexico State University, Random House, New York 1987.

[18] Xu J. and Croft W., "Cluster-Based Language Models for Distribution Retrieval," *in Proceedings of the ACM/SIGIR*, pp. 254-261, 1999.

[19] Xu R. and Wunsch II D., "Survey of Clustering Algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645-678, 2005.

[20] Zamir O. and Etzoni O., "Web Document Clustering: A Feasibility Demonstration," *in Proceedings of the 1998 ACM/SIGIR*, pp. 46-54, 1998.

Yaser Hasan is a lecturer in the Department of Computer Science at the Zarqa Private University, Jordan. He was obtained his BSc in computer science in 1985, high diploma in mathematics in 1991, both from the University of Jordan, a Masters in CIS from the Arab Academy of Finance and Banking Sciences in 2004, and currently he is a PhD candidate at the University of Bradford UK.

Mohammad Hassan is an assistant professor in the Department of Computer Science, Zarqa Private University, Jordan. He obtained his BSc in mathematics/computer science from Yarmouk University, Jordan in 1987, and his MSc in mathematics form University of Jordan in 1995. His PhD in information retrieval systems is obtained form Bradford University, UK in 2003.

Mick Ridely is a lecturer in the Department of Computing. He is currently head of Department. His research work is in the area of databases, particularly bibliographic applications including BOPAC. He was a member of the Joint Steering Committee for Revision of Anglo-American Cataloging Rules: Format Variation Working Group.